

# LẬP TRÌNH MẠNG CĂN BẢN

Biên soạn: ThS. Đỗ Thị Hương Lan



TRƯỜNG ĐH CÔNG NGHỆ THÔNG TIN - ĐHQG-HCM  
**KHOA MẠNG MÁY TÍNH & TRUYỀN THÔNG**  
FACULTY OF COMPUTER NETWORK AND COMMUNICATIONS

Tầng 8 - Tòa nhà E, trường ĐH Công nghệ Thông tin, ĐHQG-HCM

## **Chương 6**

# **Lập trình với Giao thức Tầng ứng dụng HTTP**

# **Nội dung**

# **Giao thức HTTP**

# **và Ứng dụng Web**

# Nội dung chi tiết

---

- Giới thiệu về giao thức HTTP và ứng dụng Web
- HTTP Request và HTTP Response
- Lập trình Web client
- Lập trình Web server

# Nội dung chi tiết

---

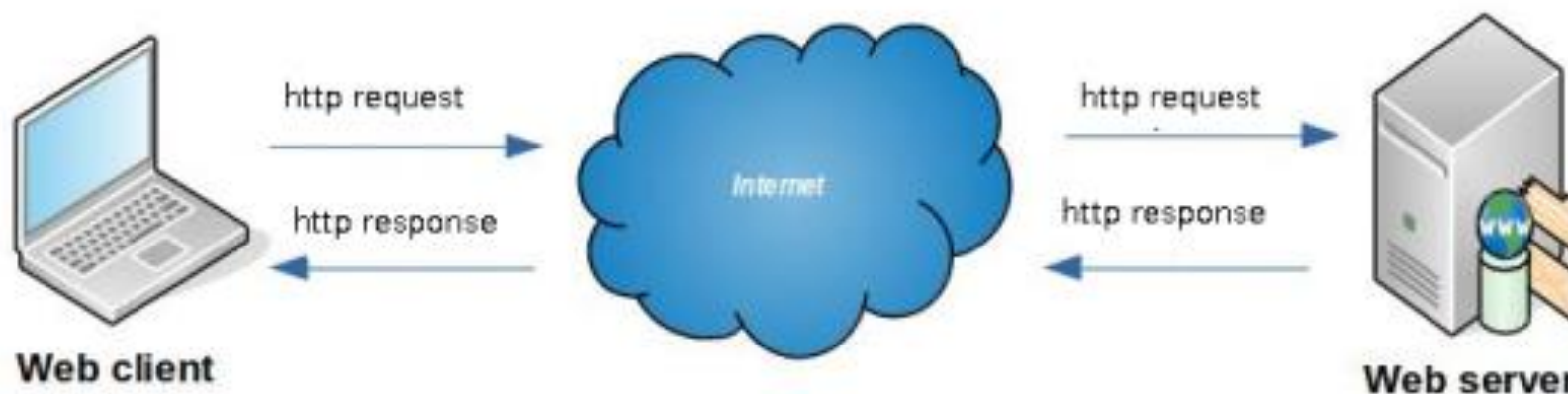
- Giới thiệu về giao thức HTTP và ứng dụng Web
- HTTP Request và HTTP Response
- Lập trình Web client
- Lập trình Web server

# Giới thiệu

- Cách lấy dữ liệu từ Web và sử dụng vào mục đích khác
- Tại sao ứng dụng cần giao tiếp với website?
  - Kiểm tra các bản cập nhật, sửa lỗi, nâng cấp
  - Lấy thông tin về dữ liệu được cập nhật hằng giờ
  - Tự động truy vấn dữ liệu từ các dịch vụ điều hành bởi bên thứ 3
  - Xây dựng search engine
  - Cache các trang web để truy cập nhanh hơn

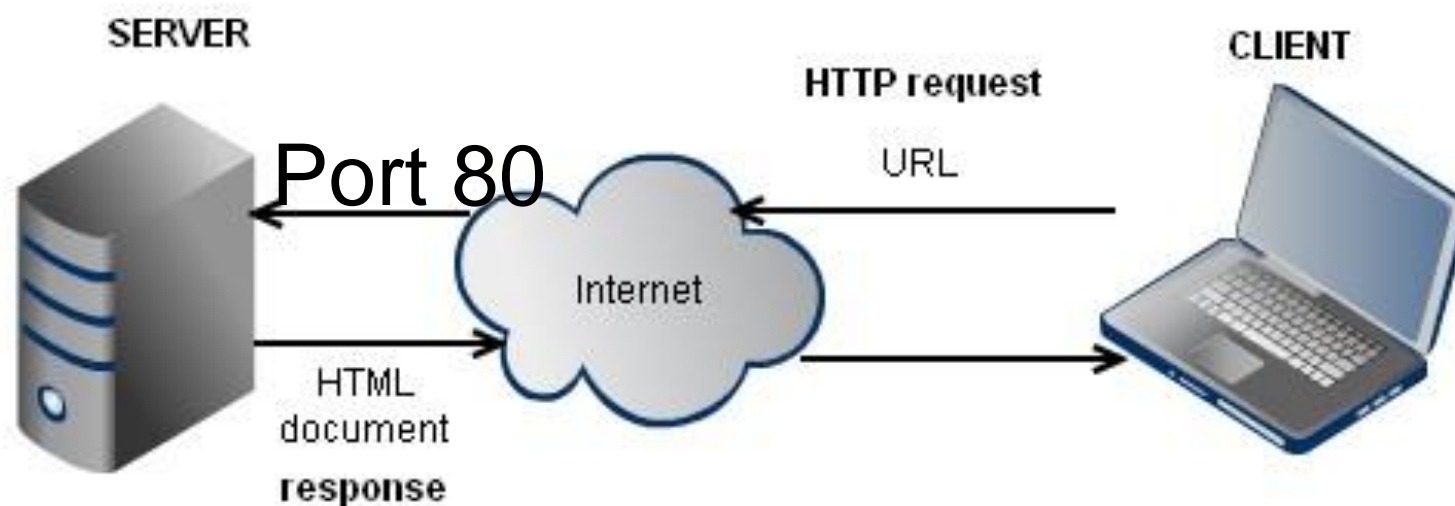
# Mô hình hoạt động

- Web Server **không phải** Website
- Web Client **thường** là trình duyệt (Browser)
- HTTP là giao thức sử dụng để giao tiếp giữa Web Server và Web Client



# Giao thức HTTP

Hoạt động trên giao thức TCP/IP port 80

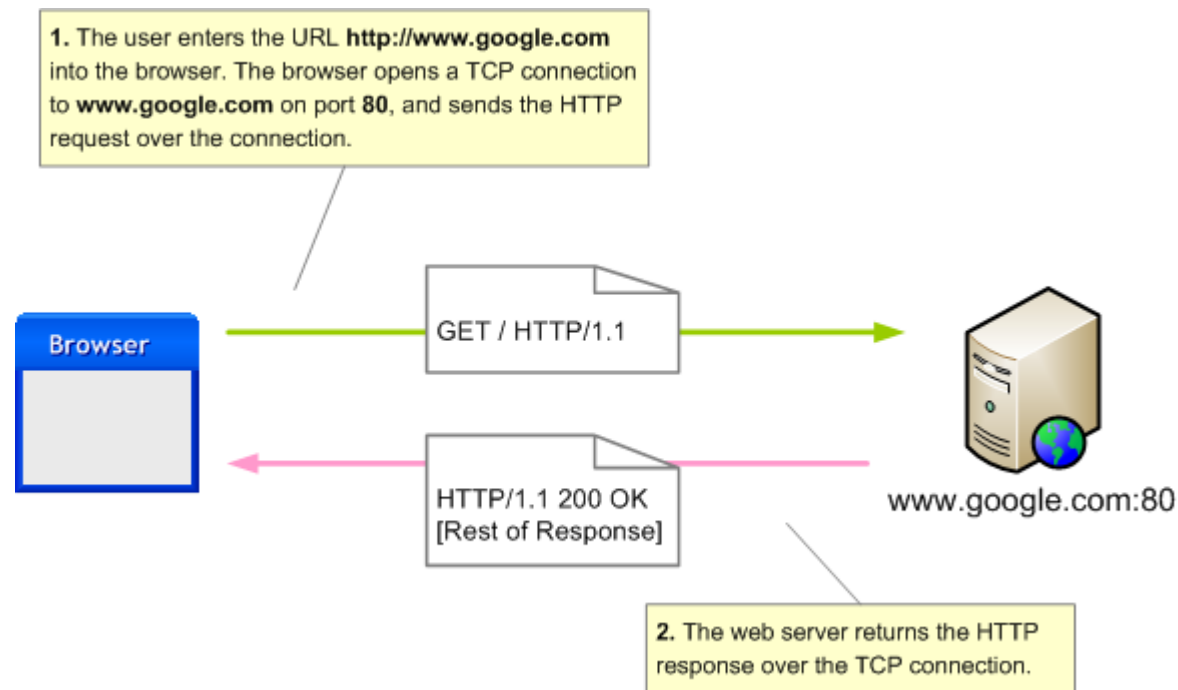




# Giao thức HTTP

Hoạt động trên giao thức TCP/IP port 80

Gồm 2 loại thông điệp: HTTP Request & HTTP Response



# Web server

- Ở khía cạnh **phần cứng**, một web server là một máy tính lưu trữ các thành phần của một website. Nó kết nối tới mạng Internet, thường có IP cố định, và thường được truy cập đến thông qua tên miền
- Ở khía cạnh **phần mềm**, một web server là một phần mềm hiểu được các URL (các địa chỉ web) và HTTP (giao thức trình duyệt của bạn sử dụng để xem các trang web), lắng nghe và xử lý các thông điệp được gửi đến từ Web Client (thường là Browser) => **Lập trình viên quan tâm đến**

# Web server

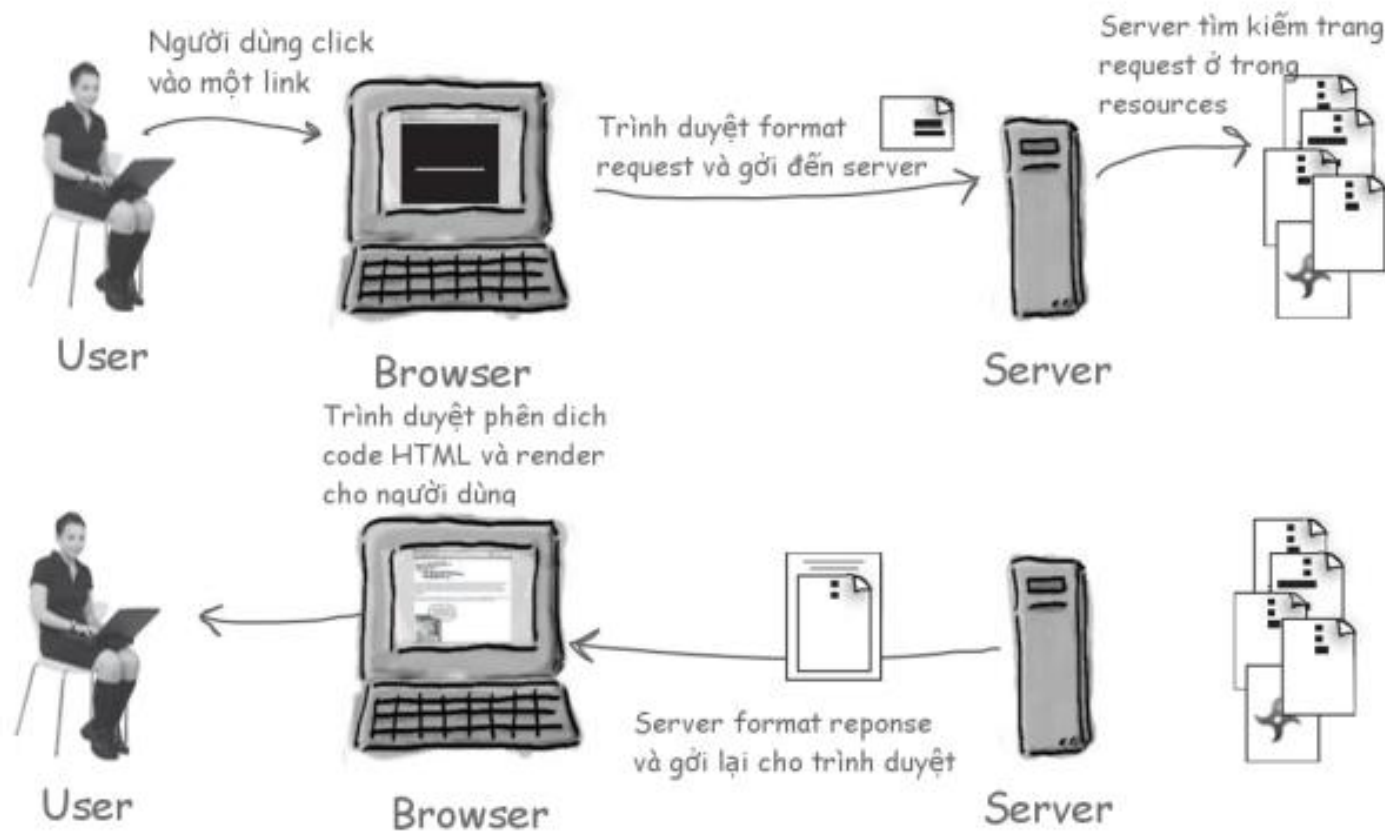
- **Chức năng chính:**
  - Lưu trữ
  - Xử lý
  - Chuyển trang web đến client
- Giao tiếp giữa client và server: HTTP
- Trang được chuyển: HTTP document, image, css, script

# Web client

---

- **Thường là Trình duyệt (Browser)**
- **Chức năng chính:**
  - Là phần mềm giúp người dùng giao tiếp với server.
  - Gửi đi những thông điệp Request dựa vào thao tác của người dùng
  - Tiếp nhận thông điệp Response
  - Biên dịch HTML và hiển thị (rendering) trang web cho user.

# Web client



# Nội dung chi tiết

---

- Giới thiệu về giao thức HTTP và ứng dụng Web
- **HTTP Request và HTTP Response**
- Lập trình Web client
- Lập trình Web server

# HTTP Request

- Là **thông điệp** được gửi từ Client đến Server
- Có 2 dạng phổ biến: GET và POST
- Còn có HEAD, OPTIONS, PUT, DELETE, TRACE

# HTTP request GET

Dòng yêu cầu  
(các lệnh GET, POST,  
HEAD)

Các dòng header

Ký tự xuống dòng,  
về đầu dòng mới chỉ  
điểm cuối cùng  
của thông điệp

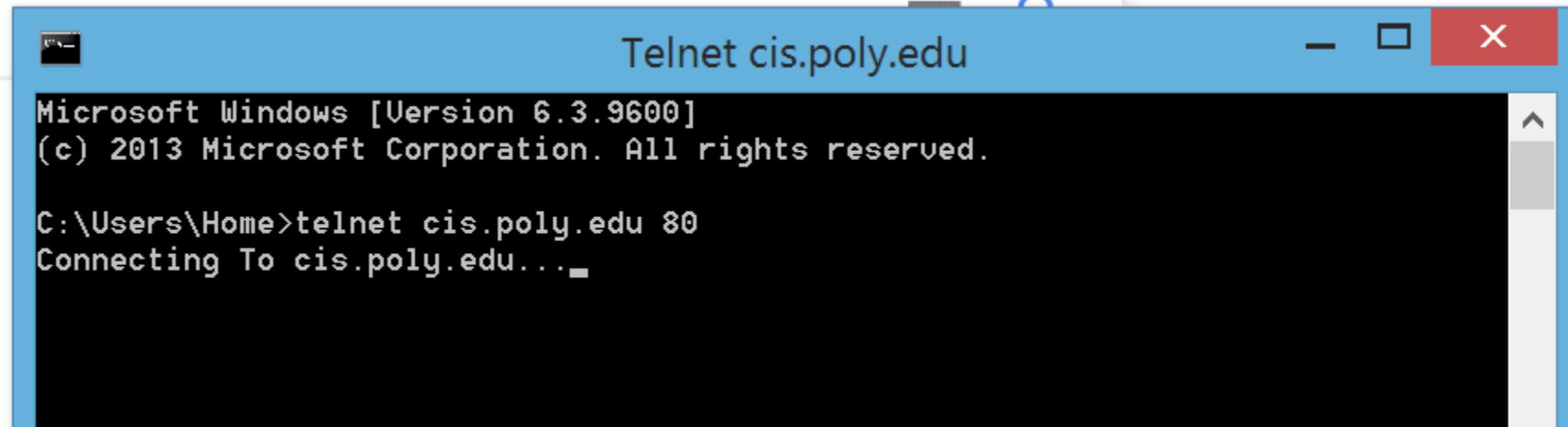
```
GET /index.html HTTP/1.1\r\n
Host: www-net.cs.umass.edu\r\n
User-Agent: Firefox/3.6.10\r\n
Accept: text/html,application/xhtml+xml\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
Keep-Alive: 115\r\n
Connection: keep-alive\r\n
\r\n
```

Ký tự xuống dòng  
Ký tự về đầu dòng



# VÍ DỤ: HTTP request với Telnet

- Mở kết nối tới port 80 của 1 website



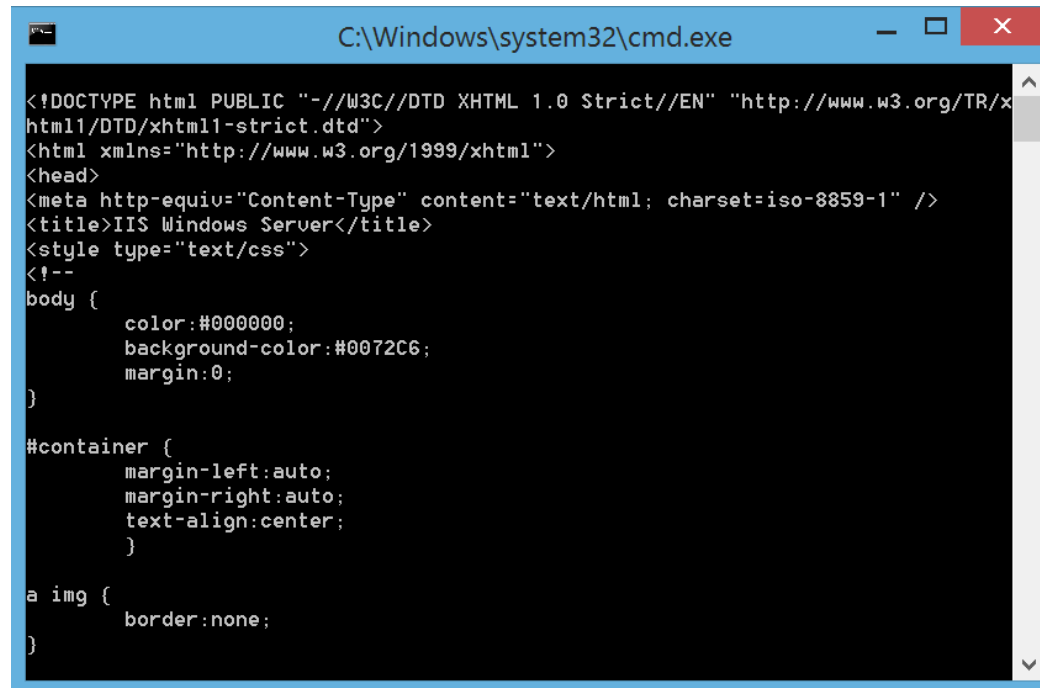
```
Telnet cis.poly.edu
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Home>telnet cis.poly.edu 80
Connecting To cis.poly.edu..._
```

- Gõ **GET /**

# VÍ DỤ: HTTP request với Telnet

- Gõ **GET /** => Gửi thông điệp HTTP Request GET
- Kết quả nhận được:



```
C:\Windows\system32\cmd.exe

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>IIS Windows Server</title>
<style type="text/css">
<!--
body {
    color:#000000;
    background-color:#0072C6;
    margin:0;
}

#container {
    margin-left:auto;
    margin-right:auto;
    text-align:center;
}

a img {
    border:none;
}
```

# HTTP Request headers

HTTP header	Ý nghĩa
Accept	Xác định kiểu MIME nào được chấp nhận cho response. */* chỉ thị cho chấp nhận tất cả. Type/* chỉ thị các kiểu con của type đó.
Accept-Charset	Xác định các character set được chấp nhận trong response. Nếu client phát Accept-Charset: iso-8859-5 thì server biết rằng client không hiển thị được các ký tự tiếng Nhật

# HTTP Request headers (tiếp theo)

HTTP header	Ý nghĩa
Accept-Encoding	Xác định client có thể quản lý dữ liệu nén. Trong ví dụ trên cho biết trình duyệt hiểu được chuẩn nén GZIP
Accept-Language	Xác định ngôn ngữ thích hợp cho người dùng, có thể liên quan vị trí địa lý, ví dụ en-gb chỉ thị United Kingdom
Authorization	Cung cấp chứng thực giữa client và server
Host	Chỉ địa chỉ IP của server có thể dùng, có thể khác với địa chỉ IP đích nếu phải đi qua proxy.
If-Modified-Since	Cho biết trang web không cần trả về nếu không có thay đổi từ ngày xác định. Điều này cho phép cơ chế cache làm việc hiệu quả hơn. Ví dụ: If-Modified-Since: Sat, 29 Oct 1994 19:43:31 GMT

# HTTP Request headers (tiếp theo)

HTTP header	Ý nghĩa
Proxy-Authorization	Cung cấp chứng thực giữa client và proxy
Range	Cung cấp cơ chế lấy một phần trang web dựa trên vùng byte. Ví dụ: bytes=500-600,601-999
Referer	Cho biết trang client vừa xem
TE	Transfer encoding (TE) cho biết phần mở rộng encoding có thể chấp nhận
User-Agent	Chỉ kiểu trình duyệt client đang dùng
Content-Type	Dùng trong các POST request, chỉ kiểu MIME của dữ liệu được post lên, thông thường là application/x-www-form-urlencoded
Content-Length	Dùng trong các POST request, chỉ độ dài của dữ liệu (đi sau 2 dòng trống)

# HTTP POST Request

---

POST / HTTP/1.1

Content-Type: application/x-www-form-urlencoded

Content-Length: 17

myField=some+text

# HTTP Response

- Là **thông điệp** được được phản hồi từ Server về cho Client
- Khi server nhận được một HTTP request, trả về trang yêu cầu cùng với HTTP header
- Đi kèm **mã trạng thái (status code)**

# HTTP Response 200 OK

Dòng trạng thái  
(giao thức  
mã trạng thái  
cụm từ trạng thái)

Các dòng  
header

Dữ liệu, ví dụ,  
tập tin HTML  
được yêu cầu

```
HTTP/1.1 200 OK\r\n
Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n
máy chủ: Apache/2.0.52 (CentOS)\r\n
Last-Modified: Tue, 30 Oct 2007 17:00:02 GMT\r\n
ETag: "17dc6-a5c-bf716880"\r\n
Accept-Ranges: bytes\r\n
Content-Length: 2652\r\n
Keep-Alive: timeout=10, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html; charset=ISO-8859-1\r\n
\r\n
data data data data data ...
```



# HTTP Response – Các trường Header

HTTP response header	Ý nghĩa
ETag	Dùng kết hợp với If-suffixed HTTP requests
Location	Dùng để điều hướng (redirect) sang trang web khác, kết hợp với HTTP 3xx responses
Proxy-Authenticate	Cung cấp chứng thực giữa client và proxy
Server	Chỉ phiên bản và vendor của server. Ví dụ: IIS chạy trên WindowsXP
WWW-Authenticate	Cung cấp chứng thực giữa client và proxy
Content-Type	Chỉ kiểu MIME của nội dung trả về. Ví dụ: HTML
Content-Length	Chỉ độ dài của dữ liệu (đi sau 2 dòng trống). Server sẽ đóng kết nối sau khi gửi tất cả dữ liệu, do đó không cần thiết xử lý lệnh này
Set-Cookie	Thiết lập một cookie trên client. Cookie là một file nhỏ ghi trên client. Mỗi cookie có tên và giá trị. Ví dụ: tên cookie là ASPSESSIONID

# HTTP Response – Mã trạng thái (Status Code)

HTTP response code range	Ý nghĩa
100–199	Thông tin: Request đã được nhận, tiếp tục xử lý
200–299	Thành công: Thao tác đã nhận thành công, hiểu được và chấp nhận
300–399	Điều hướng: Phải thêm thao tác để hoàn thành request
400–499	Lỗi client: Các mã trạng thái này chỉ ra rằng có khả năng xảy ra lỗi trong yêu cầu khiến máy chủ không thể xử lý nó
500-599	Lỗi server: Server không thể đáp ứng một request hợp lệ

# Nội dung chi tiết

---

- Giới thiệu về giao thức HTTP và ứng dụng Web
- HTTP Request và HTTP Response
- **Lập trình Web client**
- Lập trình Web server

# Lập trình Web client

- Thực hiện 2 chức năng:
  - **Giao tiếp** với server
    - Gửi đi những thông điệp HTTP Request
    - Tiếp nhận thông điệp HTTP Response
  - **Biên dịch** HTML và hiển thị (rendering) nội dung (thường là **Web Browser**)

# Để xây dựng Web client

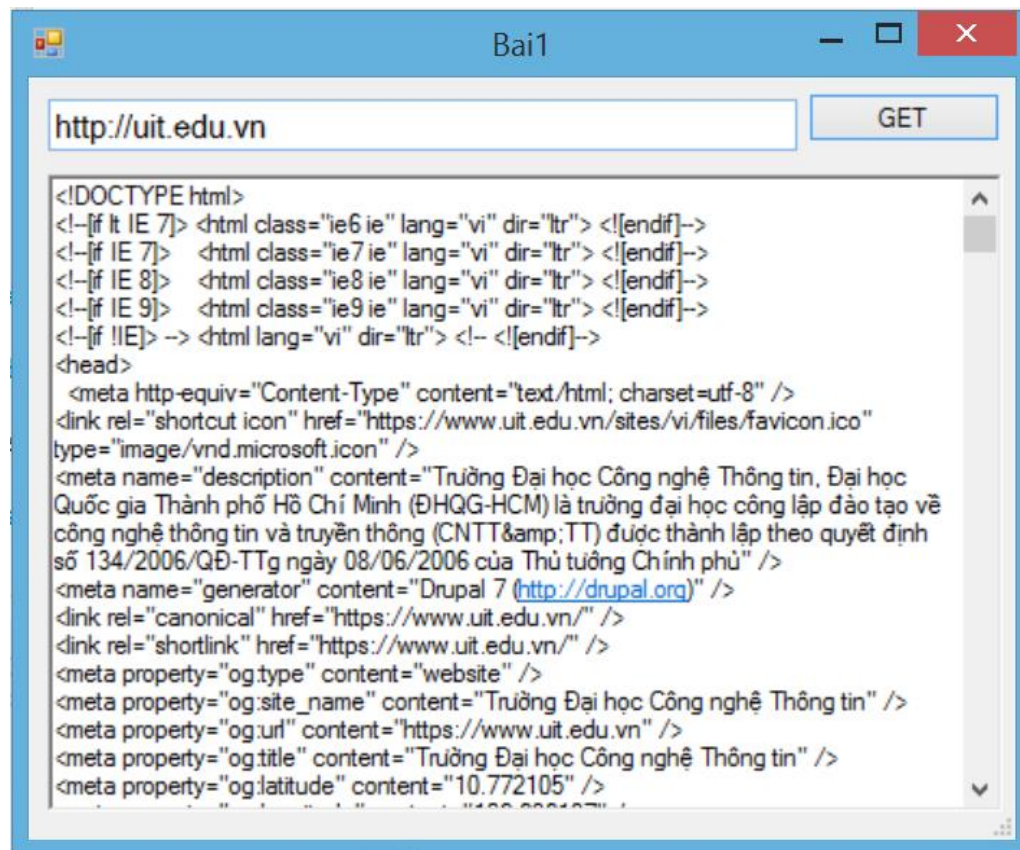
- **HttpWebRequest**: tốt cho kiểm soát
- **WebClient**: đơn giản và ngắn gọn
- **HttpClient**: mạnh mẽ nhất, cho cả hai tính năng trên môi trường .NET 4.5
- **TCPClient**: cơ bản, tự định nghĩa

# HttpWebRequest & WebClient & HttpClient

HTTPWebRequest	WebClient	HttpClient
Tất cả version	Tất cả version	Từ .NET Framework 4.5
Class tiêu chuẩn	Bản tóm tắt, đơn giản của HTTPWebRequest	Mạnh mẽ nhất

# HttpWebRequest - Ví dụ

- Viết hàm bắt sự kiện nút **GET** và gọi hàm **getHTML**, hiển thị thông điệp từ hàm **getHTML** vào vùng hiển thị (giả sử dùng RichTextBox)



# HttpWebRequest - Ví dụ

```
private string getHTML(string szUrl)
{
    // Khởi tạo 1 HttpWebRequest cho 1 URL.
    HttpWebRequest request = (HttpWebRequest)WebRequest.Create(szUrl);
    // Nhận thông điệp phản hồi.
    HttpWebResponse response = (HttpWebResponse)request.GetResponse();
    // Đón nhận stream
    Stream dataStream = response.GetResponseStream();
    StreamReader reader = new StreamReader(dataStream);
    // Đọc nội dung
    string responseFromServer = reader.ReadToEnd();
    response.Close();
    return responseFromServer;
}
```



# HttpWebRequest

Phương thức hoặc thuộc tính	Ý nghĩa
Accept	Lấy ra hoặc thiết lập giá trị của Accept HTTP header. Kiểu String
AllowAutoRedirect	Lấy ra hoặc thiết lập giá trị boolean cho biết có request đi sau các response điều hướng (3xx) hay không
ContentLength	Lấy ra hoặc thiết lập Content-length HTTP header
ContentType	Lấy ra hoặc thiết lập Content-type HTTP header
CookieContainer	Lấy ra hoặc thiết lập các cookie liên kết với request. Ví dụ: CookieContainer.getCookies["name"].ToString().
Headers	Lấy ra một tập string chứa trong HTTP header. Ví dụ: Headers["Content-Type"].ToString().
Method	Lấy ra hoặc thiết lập phương thức dành cho request. Có thể thiết lập là GET, HEAD, POST, PUT, DELETE, TRACE, OPTIONS

# HttpWebRequest

Phương thức hoặc thuộc tính	Ý nghĩa
Proxy	Lấy ra hoặc thiết lập thông tin Proxy cho request. Trả về WebProxy
Referer	Lấy ra hoặc thiết lập giá trị của Referer HTTP header. Trả về String
RequestUri	Lấy ra URI gốc của request. Ví dụ: RequestURI.ToString()
Timeout	Lấy ra hoặc thiết lập giá trị Timeout. Ví dụ: Timeout=(int) new TimeSpan(0,0,30).TotalMilliseconds
TransferEncoding	Lấy ra hoặc thiết lập giá trị giá trị của Transfer-encoding HTTP header. Trả về String
UserAgent	Lấy ra hoặc thiết lập giá trị giá trị của User-agent HTTP header. Trả về String
GetResponse	Trả về một webResponse từ tài nguyên Internet

# Class `HttpWebResponse`

Phương thức hoặc thuộc tính	Ý nghĩa
<code>ContentEncoding</code>	Lấy phương pháp dùng để mã hóa nội dung của response. Trả về kiểu <code>String</code>
<code>ContentLength</code>	Độ dài của nội dung trả về bởi request, kiểu <code>Long</code>
<code>ContentType</code>	Nội dung của response, kiểu <code>String</code>
<code>Cookies</code>	Lấy ra hoặc thiết lập các cookie liên kết với request. Ví dụ: <code>Cookies["name"].ToString()</code>
<code>Headers</code>	Lấy ra các header liên kết với response này từ server. Ví dụ: <code>Headers["Content-Type"].ToString()</code> .

# HttpWebResponse

Phương thức hoặc thuộc tính	Ý nghĩa
ResponseUri	Lấy ra phần URI của tài nguyên Internet đã được đáp ứng bởi request. Ví dụ: RequestURI.ToString().
Server	Lấy ra tên của server nào gửi response, kiểu String
StatusCode	Lấy ra trạng thái của response. Trả về kiểu liệt kê HttpStatusCode
GetResponseHeader	Lấy ra nội dung header xác định đã được trả về với response. Kiểu String
GetResponseStream	Lấy ra stream dùng để đọc phần thân của response. Kiểu stream

## Posting data: ví dụ

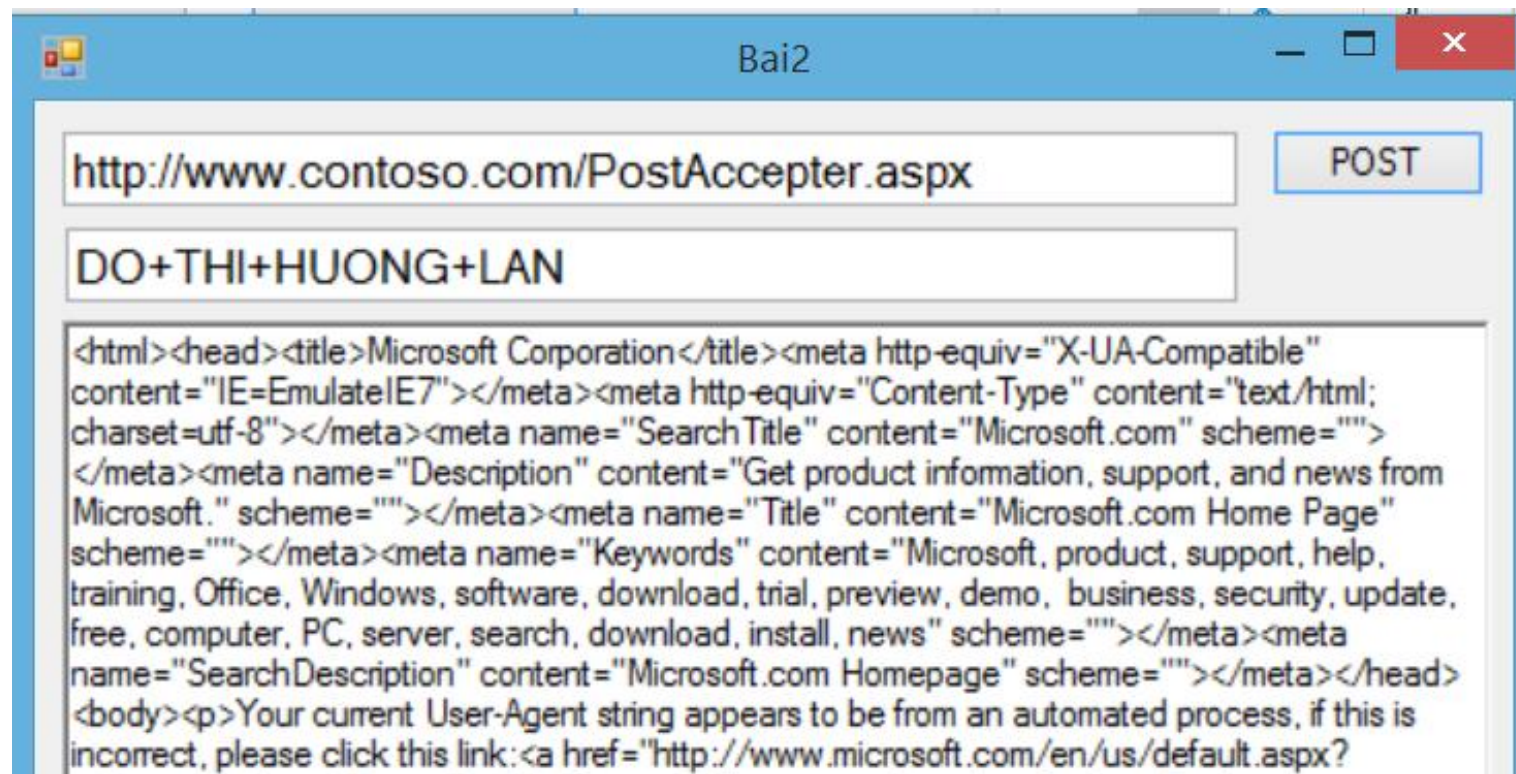
---

**Sử dụng Class WebRequest (Lớp trừu tượng) để gửi dữ liệu thông qua HTTP Request:**

**Tham khảo thêm tại:**

<https://docs.microsoft.com/en-us/dotnet/framework/network-programming/how-to-send-data-using-the-webrequest-class>

# Posting data: ví dụ



The screenshot shows a web browser window titled "Bai2". The address bar contains the URL "http://www.contoso.com/PostAcceptor.aspx". To the right of the address bar is a "POST" button. Below the address bar is a text input field containing the string "DO+THI+HUONG+LAN". Below the input field is a text area displaying the HTML response from the server. The response is an HTML document from Microsoft Corporation, with a title "Microsoft Corporation" and a meta tag "X-UA-Compatible" set to "IE=EmulateIE7". The response also includes a "SearchTitle" meta tag with the value "Microsoft.com", a "Description" meta tag with the value "Get product information, support, and news from Microsoft.", a "Title" meta tag with the value "Microsoft.com Home Page", a "Keywords" meta tag with the value "Microsoft, product, support, help, training, Office, Windows, software, download, trial, preview, demo, business, security, update, free, computer, PC, server, search, download, install, news", and a "SearchDescription" meta tag with the value "Microsoft.com Homepage". The body of the response contains a paragraph stating: "Your current User-Agent string appears to be from an automated process, if this is incorrect, please click this link:

# WebClient - Downloading

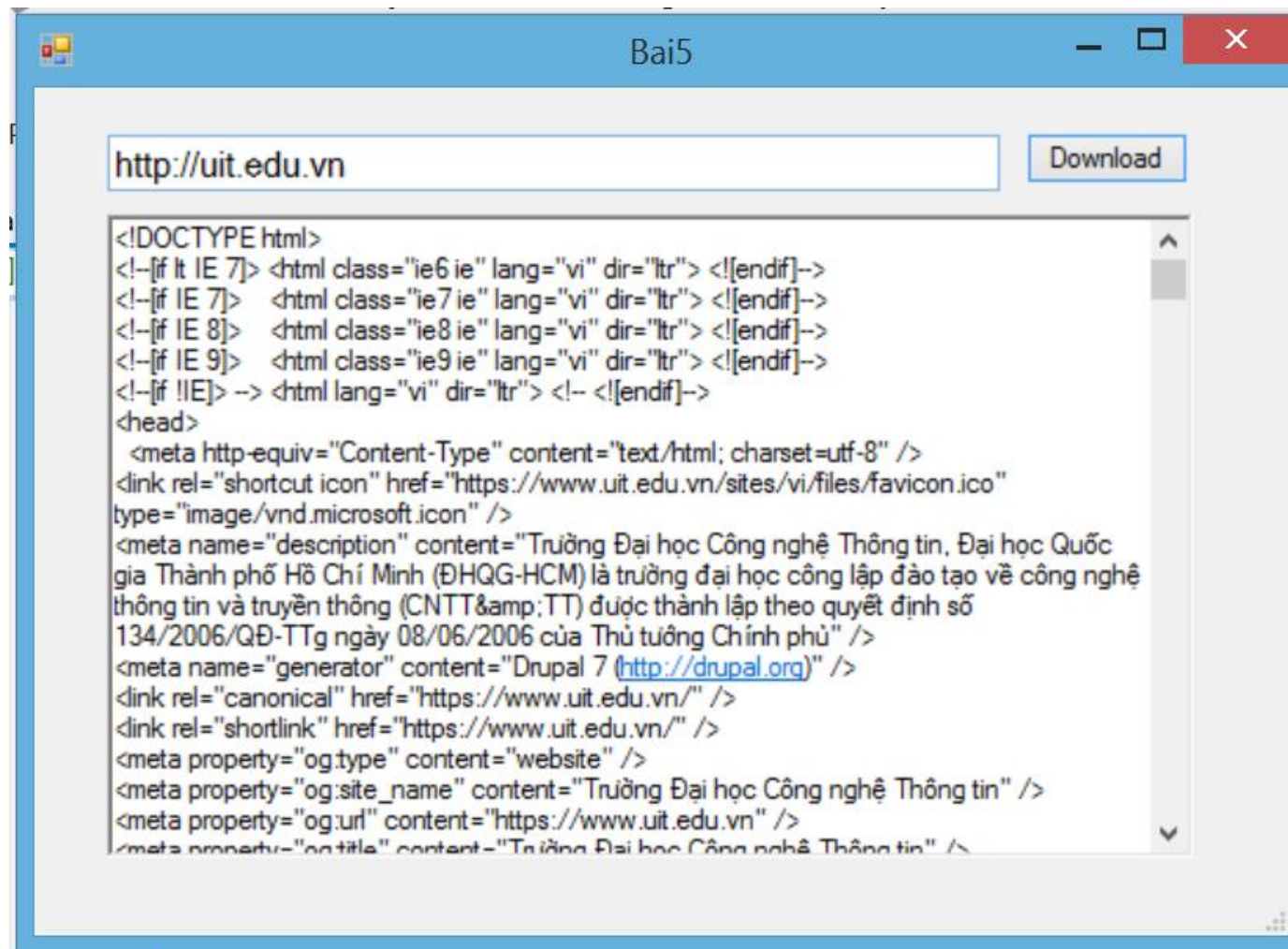
- Class WebClient cung cấp 1 số phương thức để lấy thông tin từ web server:
  1. DownloadData(): lấy dữ liệu vào một mảng byte từ URI
  2. DownloadFile(): lấy dữ liệu vào một file cục bộ từ URI
  3. OpenRead(): mở stream read-only để lấy dữ liệu từ URI
  4. Xem thêm tại <https://docs.microsoft.com/en-us/dotnet/api/system.net.webclient?view=netframework-4.8>

# Minh họa DownloadData

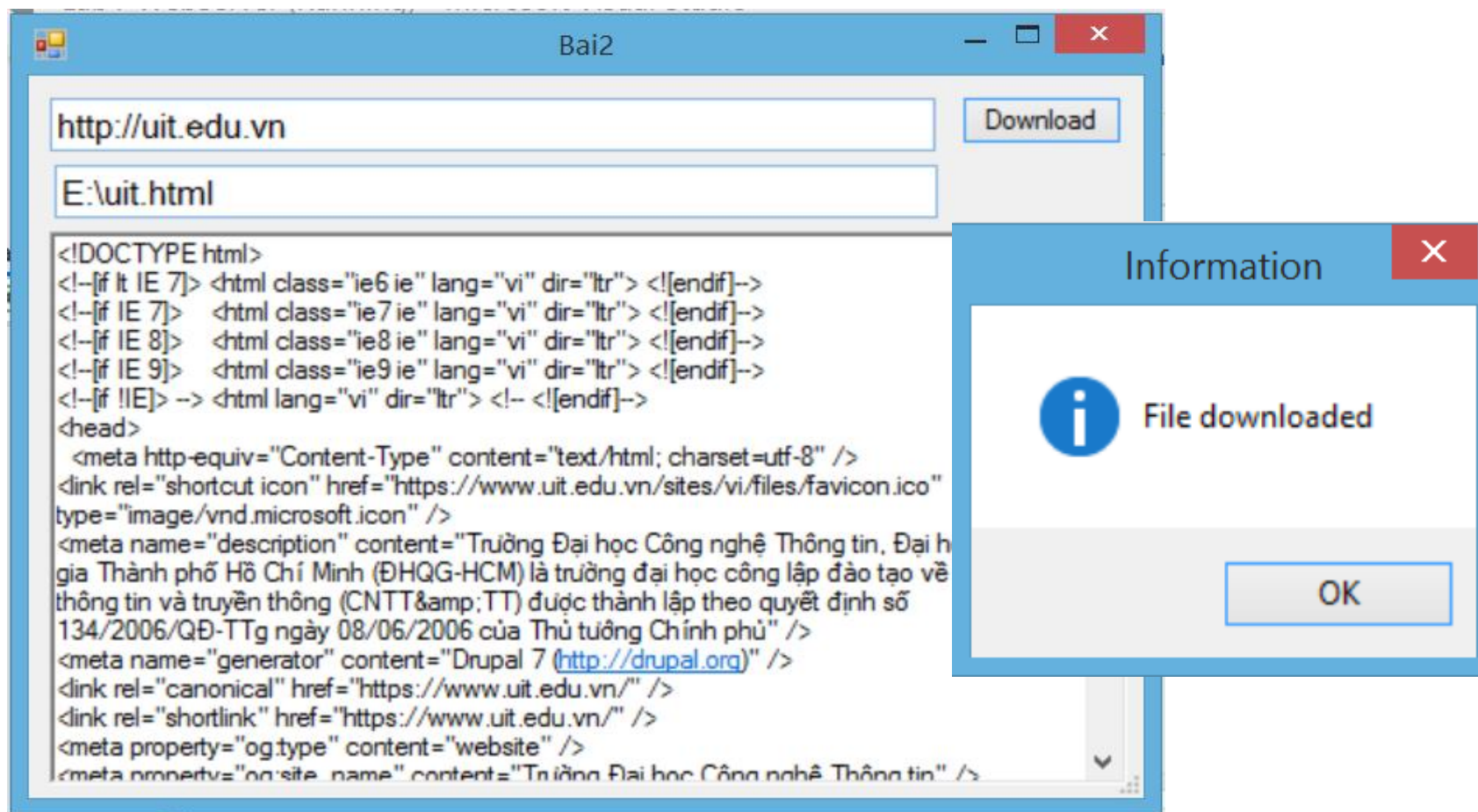
```
private void button1_Click(object sender, EventArgs e)
{
    if (txtUrl.Text.Trim() == "")
    {
        MessageBox.Show("Please input URL", "Warning", MessageBoxButtons.OK,
            MessageBoxIcon.Warning);
        return;
    }
    string url = txtUrl.Text.Trim();
    WebClient myClient = new WebClient();
    try
    {
        byte[] response = myClient.DownloadData(url);
        richTextBox1.Text = Encoding.UTF8.GetString(response);
    }
    catch (WebException wex)
    {
        richTextBox1.Text = wex.Message;
    }
}
```



# Minh họa DownloadData



# Minh họa DownloadFile



# Minh họa DownloadFile

1 reference

```
private void button1_Click(object sender, EventArgs e)
{
    string url = txtUrl.Text.Trim();
    string fileurl = txtDesFile.Text.Trim();
    WebClient myClient = new WebClient();
    Stream response = myClient.OpenRead(url);
    myClient.DownloadFile(url, fileurl);
    MessageBox.Show("File downloaded", "Information",
        MessageBoxButtons.OK, MessageBoxIcon.Information);

    StreamReader reader = new StreamReader(response);
    string responseFromServer = reader.ReadToEnd();
    richTextBox1.Text = responseFromServer;
    response.Close();
}
```

# WebClient: Property

- Class WebClient cung cấp thuộc tính ResponseHeaders để lấy thông tin các trường trong HTTP header.

```
byte[] response = myClient.DownloadData(url);  
richTextBox1.Text = Encoding.UTF8.GetString(response);  
WebHeaderCollection whc = myClient.ResponseHeaders;
```

# WebClient: property

Bai5

http://nc.uit.edu.vn

Download

```
<!doctype html >
<!--[if IE 8]> <html class="ie8" lang="en"> <![endif-->
<!--[if IE 9]> <html class="ie9" lang="en"> <![endif-->
<!--[if gt IE 8]><!--> <html lang="vi-VN"> <!--<![endif-->
<head>
  <title>Trang chủ - NC - Khoa Mạng máy tính và Truyền thông - UIT</title>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="pingback" href="https://nc.uit.edu.vn/xmlrpc.php" />
  <link rel="icon" type="image/png" href="https://nc.uit.edu.vn/wp-
content/uploads/2014/12/logo_mmt-nth.png">
  <!-- This site is optimized with the Yoast SEO plugin v13.1 -
https://yoast.com/wordpress/plugins/seo/ -->
  <meta name="robots" content="max-snippet:-1, max-image-preview:large, max-
video-preview:-1"/>
  <link rel="canonical" href="https://nc.uit.edu.vn/" />
  <meta property="og:locale" content="vi_VN" />
  <meta property="og:type" content="website" />
  <meta property="og:title" content="Trang chủ - NC - Khoa Mạng máy tính và
Truyền thông - UIT" />
  <meta property="og:url" content="https://nc.uit.edu.vn/" />
  <meta property="og:site_name" content="Khoa Mạng máy tính và Truyền thông -
UIT" />
  <meta name="twitter:card" content="summary_large_image" />
  <meta name="twitter:title" content="Trang chủ - NC - Khoa Mạng máy tính và
Truyền thông - UIT" />
```

STT	Header	Value
1	Transfer-Encoding	chunked
2	Connection	keep-alive
3	Vary	Accept-Encoding
4	Link	<https://nc.uit.e...
5	X-Frame-Options	SAMEORIGIN
6	X-Content-Type-...	nosniff
7	X-XSS-Protection	1; mode=block
8	Strict-Transport-...	max-age=63072...
9	Referrer-Policy	unsafe-url
10	Cache-Control	public, no-cache
11	Content-Type	text/html; charse...
12	Date	Tue, 14 Apr 202...
13	Server	nginx

# WebClient: Uploading

- Class WebClient sử dụng 1 số cách để upload thông tin lên web server:
  1. OpenWrite(): gửi lên dòng stream
  2. UploadData(): gửi lên dòng mảng byte
  3. UploadFile(): gửi lên dòng file
  4. UploadValues(): gửi một đối tượng NameValueCollection các data name và value lên web server

<https://docs.microsoft.com/en-us/dotnet/api/system.net.webclient?view=netframework-4.8>

# Nội dung chi tiết

---

- Giới thiệu về giao thức HTTP và ứng dụng Web
- HTTP Request và HTTP Response
- Lập trình Web client
- **Lập trình Web server**

# Lập trình Web Server

- Thực hiện chức năng:
  - **Giao tiếp** với client
    - Tiếp nhận thông điệp HTTP Request
    - Gửi thông điệp HTTP Response
  - **Xử lý đồng thời** nhiều yêu cầu
  - **Lưu trữ** tài nguyên



# Để xây dựng Web Server

- **HttpListener**: lớp hỗ trợ
- **TCPLListener**: cơ bản, tự định nghĩa

# System.Net.HttpListener

- Một trong những phương pháp tốt để hiện thực web server là sử dụng class HttpListener
- HttpListener cung cấp Http.sys có rất nhiều chức năng, như chứng thực và mã hóa SSL – nếu tự xây dựng thì tương đối khó khăn

# HttpListener – Phương thức/Thuộc tính

Phương thức hoặc thuộc tính	Mô tả
Abort / Close	Hủy bỏ hàng đợi request
BeginGetContext	Chờ đợi một client request không đồng bộ.
EndGetContext	Quản lý client request . Trả về HttpListenerContext
GetContext()	Chờ đợi một client request đồng bộ. Trả về HttpListenerContext
Star	Khởi động web server
Stop	Dừng web server
AuthenticationSchemes	Thiết lập phương pháp chứng thực giữa server và client (Basic, Digest, NTLM). Trả về AuthenticationScheme
IsListening	Xác định xem server có đang chạy hay không

# HttpListener - Ví dụ

```
// This example requires the System and System.Net namespaces.
public static void SimpleListenerExample(string[] prefixes)
{
    if (!HttpListener.IsSupported)
    {
        Console.WriteLine ("Windows XP SP2 or Server 2003 is required to use the HttpListener class.")
        return;
    }
    // URI prefixes are required,
    // for example "http://contoso.com:8080/index/".
    if (prefixes == null || prefixes.Length == 0)
        throw new ArgumentException("prefixes");

    // Create a listener.
    HttpListener listener = new HttpListener();
    // Add the prefixes.
    foreach (string s in prefixes)
    {
        listener.Prefixes.Add(s);
    }
    listener.Start();
    Console.WriteLine("Listening...");
}
```

# HttpListener - Ví dụ

```
// Note: The GetContext method blocks while waiting for a request.
HttpListenerContext context = listener.GetContext();
HttpListenerRequest request = context.Request;
// Obtain a response object.
HttpListenerResponse response = context.Response;
// Construct a response.
string responseString = "<HTML><BODY> Hello world!</BODY></HTML>";
byte[] buffer = System.Text.Encoding.UTF8.GetBytes(responseString);
// Get a response stream and write the response to it.
response.ContentLength64 = buffer.Length;
System.IO.Stream output = response.OutputStream;
output.Write(buffer,0,buffer.Length);
// You must close the output stream.
output.Close();
listener.Stop();
```

# HttpListener - Ví dụ

- **Tham khảo thêm tại:**

<https://docs.microsoft.com/en-us/dotnet/api/system.net.httplistener?view=net-6.0>